# User

Give me a python script that takes input as xml spreadsheet (2003) and outputs as excel file.

# Bing

Sure, you can use the pandas and openpyxl libraries in Python to read an XML file and write it to an Excel file. Here's a basic script:

import pandas as pd

def xml\_to\_excel(xml\_file\_path, excel\_file\_path):

# Read the XML file

data = pd.read\_xml(xml\_file\_path)

# Write the data to an Excel file

data.to\_excel(excel\_file\_path, index=False)

# Usage

xml\_to\_excel('input.xml', 'output.xlsx')

This script reads an XML file using pandas' read\_xml function and writes it to an Excel file using the to\_excel function. You can replace 'input.xml' and 'output.xlsx' with your actual file paths.

Please note that this is a simple script and might not work with complex XML files. For more complex cases, you might need to parse the XML file manually or use a more specialized library.

Also, make sure to install the necessary libraries by running pip install pandas openpyxl in your terminal before running the script.

# User

Traceback:

>>> xml\_to\_excel("D:\\tatement1709609282764.xml","D:\\jio\_usage.xlsx")

<stdin>:2: FutureWarning: Passing literal xml to 'read\_xml' is deprecated and will be removed in a future version. To read from a literal string, wrap it in a 'StringIO' object.

Traceback (most recent call last):

File "<stdin>", line 1, in <module>

File "<stdin>", line 2, in xml\_to\_excel

File "D:\Programs\Msys2\home\juhi123\.venv\Lib\site-packages\pandas\io\xml.py", line 1160, in read\_xml

return \_parse(

^^^^^^^

File "D:\Programs\Msys2\home\juhi123\.venv\Lib\site-packages\pandas\io\xml.py", line 830, in \_parse

raise ImportError("lxml not found, please install or use the etree parser.")

ImportError: lxml not found, please install or use the etree parser.

# Bing

The error message indicates that the lxml library is not installed in your Python environment. The lxml library is required by pandas to parse XML files.

You can install it using pip, which is a package manager for Python. Here's how you can install it:

pip install lxml

After installing lxml, you should be able to run the script without encountering the ImportError. If you still face any issues, feel free to ask. Happy coding! 😊

# User

I think the xml spreadsheet is bit complex. So change the script to capture data between all <Table> and </Table> tags. Mind you, there are multiple tables. Merge them and create soreadsheet with only one tab.

# Bing

Sure, I understand that your XML file contains multiple tables and you want to merge them into a single Excel sheet. Here's a Python script using lxml and pandas that should do the job:

import pandas as pd

from lxml import etree

def xml\_to\_excel(xml\_file\_path, excel\_file\_path):

# Parse the XML file

tree = etree.parse(xml\_file\_path)

root = tree.getroot()

# Find all 'Table' elements

tables = root.findall('.//Table')

# Parse each table into a DataFrame and append to a list

dfs = [pd.DataFrame(table.text for table in tbl) for tbl in tables]

# Concatenate all DataFrames in the list

data = pd.concat(dfs, ignore\_index=True)

# Write the data to an Excel file

data.to\_excel(excel\_file\_path, index=False)

# Usage

xml\_to\_excel('input.xml', 'output.xlsx')

This script finds all 'Table' elements in the XML file, parses each one into a pandas DataFrame, and then concatenates them into a single DataFrame. It then writes this DataFrame to an Excel file.

Please note that this script assumes that all tables in the XML file have the same structure (i.e., the same columns in the same order). If this is not the case, you may need to modify the script to handle the differences between tables.

Also, don't forget to install the necessary libraries by running pip install pandas lxml openpyxl in your terminal before running the script. If you still face any issues, feel free to ask. Happy coding! 😊

# User

Some problem in your script. I got only 0 in cell A1 of excel file generated. See my structure (partial) of xml:

<Table>

<TR>

<TH>Jio Number </TH>

<TH>6353676309 </TH>

</TR>

<TR>

<TH>Ms. Juhi Dhawal Bakshi B-202 Forum Complex </TH>

<TD>Account Number </TD>

<TD>200070724472 </TD>

</TR>

<TR>

<TH>Subhanpura High Tension Road Opp Madhur Milan Society </TH>

<TD/>

<TD/>

</TR>

<TR>

<TH>Vadodara Vadodara Gujarat India 390023 </TH>

<TD>Statement Date </TD>

<TD>05-Mar-2024 </TD>

</TR>

<TR>

<TH/>

<TD>: Statement Time </TD>

<TD>08:57:55 </TD>

</TR>

<TR>

<TH/>

<TD>: </TD>

<TD/>

</TR>

</Table>

<P>Security Deposit ` 0.00 : </P>

<P>Jio Membership : Prime Registered Mobile : +916353676309 | Aadhaar Number : XXXX XXXX 8086 | E-mail : bakshi.juhi@gmail.com </P>

<Part>

<H5>Account Usage and Recharge Statement from 05-Feb-2024 to 05-Mar-2024 </H5>

<P>Refer following pages for details of charges </P>

<Table>

<TR>

<TH>Amount (`) </TH>

</TR>

<TR>

<TH>1 </TH>

<TD>Plan Recharges </TD>

<TD/>

<TD/>

<TD/>

<TD>0.00 </TD>

</TR>

<TR>

<TH>2 </TH>

<TD>Usage Charges </TD>

<TD>Volume / Duration </TD>

<TD/>

<TD>Amount(`) </TD>

<TD/>

</TR>

<TR>

<TH>Off-Net / Premium / ISD & IR Calling SMS </TH>

<TD>00:00:00 0 </TD>

<TD/>

<TD>0.00 0.00 </TD>

<TD/>

</TR>

<TR>

<TH>Data 4G Data/WIFI 5G Data </TH>

<TD>0.000 GB 0.000 GB </TD>

<TD/>

<TD>0.00 0.00 </TD>

<TD/>

</TR>

<TR>

<TH>VAS </TH>

<TD>0 </TD>

<TD/>

<TD>0.00 </TD>

<TD>0.00 </TD>

</TR>

<TR>

<TH>3 Add On Packs </TH>

<TD/>

<TD/>

<TD/>

<TD>48.00 </TD>

</TR>

<TR>

<TH>Total Usage Summary Voice Local On-Net/Off-Net(LL) Off-Net(Mobile) Voice STD On-Net/Off-Net(LL) Off-Net(Mobile) Premium / ISD & IR Calling SMS Data 4G Data/WIFI 5G Data VAS </TH>

<TD>Total Usage 00:12:17 00:00:00 01:19:53 00:00:00 00:00:00 14 17.371 GB 17.371 GB 0.000 GB 0 </TD>

<TD>Chargeable Usage 00:00:00 00:00:00 00:00:00 00:00:00 00:00:00 0 0.000 GB 0.000 GB 0.000 GB 0 </TD>

<TD>Chargeable Amount (`) 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 </TD>

<TD/>

</TR>

<TR>

<TH>4 </TH>

<TD>Other Purchases </TD>

<TD/>

<TD/>

<TD/>

<TD>0.00 </TD>

</TR>

<TR>

<TH>5 </TH>

<TD>Credit & Adjustments </TD>

<TD/>

<TD/>

<TD/>

<TD>0.00 </TD>

</TR>

</Table>

<Table>

<TR>

<TH>Recharge Option </TH>

</TR>

<TR>

<TH>Jio Self Care </TH>

<TD>Jio Store </TD>

</TR>

<TR>

<TH>Credit Card / </TH>

<TD>My Jio & Jio.com </TD>

<TD>Cash / Credit / Debit </TD>

</TR>

</Table>

# Bing

I see, the XML structure is indeed more complex than initially assumed. The tables in your XML file have different structures and the data is nested within <TR> and <TH> or <TD> tags.

Here's a revised script that should handle this structure:

import pandas as pd

from lxml import etree

def xml\_to\_excel(xml\_file\_path, excel\_file\_path):

# Parse the XML file

tree = etree.parse(xml\_file\_path)

root = tree.getroot()

# Find all 'Table' elements

tables = root.findall('.//Table')

# Initialize an empty DataFrame

data = pd.DataFrame()

# Loop over each table

for table in tables:

# Find all 'TR' elements in the table

rows = table.findall('.//TR')

# Loop over each row

for row in rows:

# Find all 'TH' and 'TD' elements in the row

cols = row.findall('.//TH') + row.findall('.//TD')

# Get the text from each column

cols\_text = [col.text for col in cols]

# Append the row to the DataFrame

data = data.append(pd.Series(cols\_text), ignore\_index=True)

# Write the data to an Excel file

data.to\_excel(excel\_file\_path, index=False)

# Usage

xml\_to\_excel('input.xml', 'output.xlsx')

This script should correctly parse the tables in your XML file and merge them into a single Excel sheet. It handles rows with a mix of <TH> and <TD> elements and appends each row to the DataFrame.

Please note that this script assumes that all rows in all tables have the same number of columns. If this is not the case, you may need to modify the script to handle rows with different numbers of columns.

Also, don't forget to install the necessary libraries by running pip install pandas lxml openpyxl in your terminal before running the script. If you still face any issues, feel free to ask. Happy coding! 😊

# User

Some issue when I ran:

$ py xml\_xl.py

Traceback (most recent call last):
File "D:\xml\_xl.py", line 35, in <module>
xml\_to\_excel('d:\Statement1709609282764.xml', 'd:\jio\_usage.xlsx')
File "D:\xml\_xl.py", line 29, in xml\_to\_excel
data = data.append(pd.Series(cols\_text), ignore\_index=True)
^^^^^^^^^^^
File "D:\Programs\Msys2\home\juhi123.venv\Lib\site-packages\pandas\core\generic.py", line 6296, in **getattr**
return object.**getattribute**(self, name)
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
AttributeError: 'DataFrame' object has no attribute 'append'. Did you mean: '\_append'?

# Bing

I apologize for the confusion. The append method is indeed a function of pandas DataFrame. However, it seems like there's an issue with the way it's being used.

In pandas, append is used to append rows of other DataFrame to the end of the given DataFrame, returning a new DataFrame object. Here, we are trying to append a Series to a DataFrame, which might be causing the issue.

Let's modify the script to use pd.concat for appending the Series to the DataFrame:

import pandas as pd

from lxml import etree

def xml\_to\_excel(xml\_file\_path, excel\_file\_path):

# Parse the XML file

tree = etree.parse(xml\_file\_path)

root = tree.getroot()

# Find all 'Table' elements

tables = root.findall('.//Table')

# Initialize an empty list to store DataFrames

dfs = []

# Loop over each table

for table in tables:

# Find all 'TR' elements in the table

rows = table.findall('.//TR')

# Loop over each row

for row in rows:

# Find all 'TH' and 'TD' elements in the row

cols = row.findall('.//TH') + row.findall('.//TD')

# Get the text from each column

cols\_text = [col.text for col in cols]

# Append the row to the list as a DataFrame

dfs.append(pd.DataFrame([cols\_text]))

# Concatenate all DataFrames in the list

data = pd.concat(dfs, ignore\_index=True)

# Write the data to an Excel file

data.to\_excel(excel\_file\_path, index=False)

# Usage

xml\_to\_excel('input.xml', 'output.xlsx')

This script should correctly parse the tables in your XML file and merge them into a single Excel sheet. It handles rows with a mix of <TH> and <TD> elements and appends each row to the DataFrame.

Please note that this script assumes that all rows in all tables have the same number of columns. If this is not the case, you may need to modify the script to handle rows with different numbers of columns.

Also, don't forget to install the necessary libraries by running pip install pandas lxml openpyxl in your terminal before running the script. If you still face any issues, feel free to ask. Happy coding! 😊